UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

4

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/761,564	01/21/2004	Edwin F. Barry	800.0136 (A1132 C1)	3447

27997 7590 11/23/2004

PRIEST & GOLDSTEIN PLLC
5015 SOUTHPARK DRIVE
SUITE 230
DURHAM, NC 27713-7736

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 11/23/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/761,564	BARRY ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 October 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *14-29* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *14-29* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *12 October 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 14-29 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment as received on 10/12/2004.

Claim Objections

3. Claim 19 objected to because of the following informalities: In the second to last line, please replace "a second software task" with --the second software task--. The 35 U.S.C. 112 rejection given in the previous Office Action was a mistake on the examiner's behalf. Also, it is recommended that applicant insert either --where-- or --wherein-- before "the first software task" in line 9 and before "the second software task" in line 12. Appropriate correction is required.

Maintained Rejections

4. Applicant has failed to overcome the prior art rejections set forth in the previous Office Action. Consequently, these rejections are respectfully maintained by the examiner and copied below for applicant's convenience.

Maintained Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 14-15 and 18-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panwar et al., U.S. Patent No. 5,890,008 (herein referred to as Panwar) in view of Gordon et al., U.S. Patent No. 4,135,247 (herein referred to as Gordon).

7. Referring to claim 14, Panwar has taught an array processor comprising:

a) a physical configuration of at least two processing elements. See the abstract, Fig.3, and column 4, lines 2-5.

b) Although Panwar has taught that each processing element will detect what state it is in (see Fig.3), Panwar has not explicitly taught a processor state register storing a context status bit (CSB), the CSB having a first state and a second state, each processing element operating to detect the state of the CSB. However, Gordon has taught the general idea of storing status data in a state register which represents multiple states. See column 47, lines 14-16. A person of ordinary skill in the art would have recognized that the state of the processors must be tracked somehow, and as Gordon has taught, a state register may be used to store data (bits) representing multiple states. In this case, a register in Panwar would hold bits which specify whether a particular processor is napping, sleeping, active, etc. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Panwar to include a state register for holding data which is detected by each processing element.

c) the array processor upon detection of the first state of the CSB operating in a first operating context adapted for processing a first software task where the first software task is written for the physical configuration. See the abstract and column 4, lines 2-27. Note that one application may

be divided into M threads, which would require M processing units. This would result in M processing units being made active. And, as described in part (b) above, they would be made active by setting the CSB in a respective state register. By making M units active, the M units would respond by operating in a first context.

d) the array processor upon detection of the second state of the CSB operating in a second operating context adapted for a second software task where the second software task is written for a second array processor having a different physical configuration. See the abstract and column 4, lines 2-27. Note that one application may be divided into N threads (where N differs from M), which would require N processing units. This would result in N processing units being made active. And, as described in part (b) above, each would be made active by setting the CSB in a respective state register. By making N units active, the N units would respond by operating in a first context.

8. Referring to claim 15, Panwar in view of Gordon has taught an array processor as described in claim 14. Panwar has further taught that in the first operating context, the array processor utilizes a first and a second set of register files, the first set of register files associated with one of the processing elements and the second set of register files associated with another of the processing elements. See Fig. 11 and column 14, lines 57-63, and note that if multiple processing elements are active, then the corresponding multiple register files will be utilized.

9. Referring to claim 18, Panwar in view of Gordon has taught an array processor as described in claim 14. Panwar has further taught that each processing element of the at least two processing elements has a physical identifier and a virtual identifier, wherein during the processing of the second software task, instructions are loaded to each processing element

according to its virtual identifier. See Fig.3 and note that it is inherent that each processing element has a physical identifier. That is, each element is located at a different part on the chip and would therefore each could be separated identified, physically. In addition, since the processing elements are virtual processors, they would have virtual identifiers. The instructions to be processed include a thread ID (see Fig.9), which is used to identify a virtual processor (see column 10, lines 38-44.

10. Referring to claim 19, Panwar has taught a method for providing reconfiguration of a first array processor having a first physical configuration to emulate operation of a second array processor having a second physical configuration, the method comprising:

a) providing the first array processor having at least two processor elements. See the abstract, Fig.3, and column 4, lines 2-5.

b) Although Panwar has taught that each processing element will detect what state it is in (see Fig.3), Panwar has not explicitly taught storing a context status bit (CSB), the CSB having a first state and a second state. However, Gordon has taught the general idea of storing status data in a state register which represents multiple states. See column 47, lines 14-16. A person of ordinary skill in the art would have recognized that the state of the processors must be tracked somehow, and as Gordon has taught, a state register may be used to store data (bits) representing multiple states. In this case, a register in Panwar would hold bits which specify whether a particular processor is napping, sleeping, active, etc. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Panwar to include a state register for holding data which is detected by each processing element.

c) upon detection of the first state, operating in a first operating context adapted for processing a first software task, the first software task is written for the first physical configuration. See the abstract and column 4, lines 2-27. Note that one application may be divided into M threads, which would require M processing units. This would result in M processing units being made active. And, as described in part (b) above, they would be made active by setting the CSB in a respective state register. By making M units active, the M units would respond by operating in a first context.

d) upon detection of the second state, operating in the second operating context adapted for processing a second software task, the second software task is written for the second physical configuration. See the abstract and column 4, lines 2-27. Note that one application may be divided into N threads (where N differs from M), which would require N processing units. This would result in N processing units being made active. And, as described in part (b) above, each would be made active by setting the CSB in a respective state register. By making N units active, the N units would respond by operating in a first context.

11. Referring to claim 20, Panwar in view of Gordon has taught a method as described in claim 19. Panwar has further taught that the operating in the second operating context step further comprises setting the CSB to the first state and returning to the first operating context. Although not explicitly stated by Panwar, a person of ordinary skill in the art would have recognized that the CSB bits may be changed in any way to achieve any configuration. For instance, if the first context includes 5 processing elements being active and the second context includes the same 5 processing elements being active and 1 additional processing elements being

active, then when the 1 additional processing element is dead (inactive), the CSB bit reflect that and the first context will then be achieved.

12. Referring to claim 21, Panwar in view of Gordon has taught a method as described in claim 19. Panwar has further taught that the first physical configuration comprises a 1x1 layout and the emulated second physical configuration comprises a 1x0 layout. From column 5, lines 37-40, and column 4, lines 2-27, the system can be reconfigured to include up to 64 processing elements. The amount is dependent on the number of threads. If the number of threads is 2, then there would be 2 processing elements (1x1). If there is a single thread, there would be 1 processing element (1x0).

13. Referring to claim 22, Panwar in view of Gordon has taught a method as described in claim 19. Panwar has further taught that the first physical configuration comprises a 1x2 layout and the emulated second physical configuration comprises a 1x1 layout. From column 5, lines 37-40, and column 4, lines 2-27, the system can be reconfigured to include up to 64 processing elements. The amount is dependent on the number of threads. If the number of threads is 3, then there would be 3 processing elements (1x2). If there are 2 threads, there would be 2 processing elements (1x1).

14. Referring to claim 23, Panwar in view of Gordon has taught a method as described in claim 19. Panwar has further taught that the first physical configuration comprises a 1x5 layout and the emulated second physical configuration comprises a 2x2 layout. From column 5, lines 37-40, and column 4, lines 2-27, the system can be reconfigured to include up to 64 processing elements. The amount is dependent on the number of threads. If the number of threads is 5, then

there would be 5 processing elements (1x5). If there are 4 threads, there would be 4 processing element (2x2).

15. Claims 16-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panwar in view of Gordon, as applied above, in view of Dowling, U.S. Patent No. 6,128,728.

16. Referring to claim 16, Panwar in view of Gordon has taught an array processor as described in claim 15. Panwar in view of Gordon has not explicitly taught an eventpoint mechanism to trigger storing the data contents of the first set of register files in the background while the first software task uses the second set of register files in the foreground. However, Dowling has taught such a concept. See column 4, lines 32-46, and column 5, line 14, to column 6, line 6. With such a scheme, the inactive register set would be stored in the background while the active register set would be manipulated by a processor in the foreground. This would maximize the efficiency by making use of otherwise unused external memory cycles, as disclosed in the abstract. That is, when a processor is not accessing memory, the contents of the inactive register file may be stored in the background during those unused memory cycles. Therefore, in order to increase efficiency, it would have been obvious to one of ordinary skill in the art at the time of the invention to save a second register set from memory in the background while a task is using a first register set in the foreground.

17. Referring to claim 17, Panwar in view of Gordon has taught an array processor as described in claim 15. Panwar in view of Gordon has not explicitly taught an eventpoint mechanism to trigger loading the first set of register files in the background while the first software task uses the second set of register files in the foreground. However, Dowling has

taught such a concept. See column 4, lines 32-46, and column 5, line 14, to column 6, line 6. With such a scheme, the inactive register set would be loaded in the background while the active register set would be manipulated by a processor in the foreground. This would maximize the efficiency by making use of otherwise unused external memory cycles, as disclosed in the abstract. That is, when a processor is not accessing memory, the inactive register file may be loaded in the background during those unused memory cycles. Therefore, in order to increase efficiency, it would have been obvious to one of ordinary skill in the art at the time of the invention to restore a second register set from memory in the background while a task is using a first register set in the foreground.

18. Claims 24-27 and 29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Panwar in view of Gordon, as applied above, and further in view of Keckler et al., U.S. Patent No. 5,574,939 (herein referred to as Keckler).

19. Referring to claim 24, Panwar has taught an apparatus for providing efficient sharing of programming resources in a merged very long instruction word (VLIW) sequence processor (SP) and VLIW processor element (PE) processor, the merged VLIW SP/PE processor configurable in a first merged processor configuration or in a second merged processor configuration, the apparatus comprising:

a) an SP resource file having a first set of registers. See Fig.11 and column 14, lines 57-63, and note that if multiple processing elements are active, then the corresponding multiple register files will be utilized.

b) a PE resource file having a second set of registers. See Fig.11 and column 14, lines 57-63, and note that if multiple processing elements are active, then the corresponding multiple register files will be utilized.

c) Panwar has not taught an input for receiving a VLIW presented for execution, the VLIW having at least two instructions, each instruction encoded with a different setting of an SP/PE-bit. However, Keckler has taught a system in which a VLIW comprises at least two instructions where the instructions may belong to different threads. See Fig.1. As seen from the Figure, when multiple threads are not intermixed within a single cycle, many functional units may be idle (note the empty boxes in the top half of the figure). However, when instructions are intermixed, the functional units are better utilized and more throughput is achieved (note the bottom portion of the figure). This increases the parallelism achieved by a normal VLIW instruction, which allows multiple instructions to execute in parallel. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Panwar to execute VLIW instructions which include instructions having different thread IDs (SP/PE-bits), which are used to identify the virtual processor that will execute the respective instruction.

d) Although Panwar has taught that each processing element will detect what state it is in (see Fig.3), Panwar in view of Keckler has not taught a processor state register storing a context select bit (CSB). However, Gordon has taught the general idea of storing status data which represents multiple states. See column 47, lines 14-16. A person of ordinary skill in the art would have recognized that the state of the processors must be tracked somehow, and as Gordon has taught, the data (bits) representing multiple states is stored so it can be referenced. In this case, Panwar would hold bits which specify whether a particular processor is napping, sleeping,

active, etc. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Panwar to include a store the context select bit

e) the merged VLIW SP/PE processor reading the values of the CSB and the SP/PE-bit of an instruction to select a first merged processor configuration or a second merged processor configuration when processing the instruction, the first merged processor configuration adapted for accessing at least one register from the second set of registers when processing an SP instruction. See the abstract and column 4, lines 2-27. Note that one application may be divided into M threads, which would require M processing units. This would result in M processing units being made active. And, as described in part (b) above, they would be made active by setting the CSB in a respective state register. By making M units active, the M units would respond by operating in a first context (it would operate in a second context when N threads are available, N differing from M) and each virtual processing element would execute an instruction which matches is virtual processor number (thread ID). See column 10, lines 38-41 and see Fig.9 (note each instruction has a thread ID). Finally, the second set of registers would be accessed when the processing element associated with the second set of registers is active in the first merged processor configuration.

20. Referring to claim 25, Panwar in view of Gordon and further in view of Keckler has taught an apparatus as described in claim 24. Panwar has further taught that the second merged processor configuration adapted for accessing at least one register from the first set of registers when processing an SP instruction and accessing at least one register from the second set of registers when processing a PE instruction. Clearly, the first set of registers would be accessed when the processing element associated with the first set of registers is active in the second

merged processor configuration (note that any instruction executed by one of the processing elements in the second configuration is an SP instruction). Also, the second set of registers would be accessed when the processing element associated with the second set of registers is active in the second merged processor configuration (again, note that any instruction executed by one of the processing elements in the second configuration is a PE instruction, as "PE instruction" it is just a name of an instruction).

21. Referring to claim 26, Panwar in view of Gordon and further in view of Keckler has taught an apparatus as described in claim 24. Panwar has further taught that the SP resource file is an SP register file, an SP address register file, or an SP machine state register file. See Fig.11, components 1101 and 1102 and note the register files.

22. Referring to claim 27, Panwar in view of Gordon and further in view of Keckler has taught an apparatus as described in claim 24. Panwar has further taught that the PE resource file is a PE register file, PE address register file, or a PE machine state register file. See Fig.11, components 1101 and 1102 and note the register files.

23. Referring to claim 29, Panwar in view of Gordon and further in view of Keckler has taught an apparatus as described in claim 24. Furthermore, recall from the rejection of claim 24 that it would have been obvious to modify Panwar to execute VLIW instructions since more parallelism would be achieved. Consequently, the VLIW SP processor and VLIW PE processor are indirect VLIW processors because Panwar has taught that the processors are virtual processors which would be acting as VLIW processors. Also, "indirect VLIW" is just a name for a processor, as the claim does not say how an indirect VLIW processor differs from a normal processor. Therefore, any processor would read on this.

24. Claim 28 is rejected under 35 U.S.C. 103(a) as being unpatentable over Panwar in view of Gordon in view of Keckler, as applied above, and further in view of Bapst et al., U.S. Patent No. 6,327,650 (herein referred to as Bapst).

25. Referring to claim 28, Panwar in view of Gordon and further in view of Keckler has taught an apparatus as described in claim 24.

a) Furthermore, it had been previously established that it would have been obvious to modify Panwar to execute VLIW instructions. Consequently, it is inherent that Panwar would include at least two execution units associated with the at least two instructions in the VLIW. That is, a VLIW instruction comprises multiple instructions which are executed in parallel. In order to execute these multiple instructions in parallel, multiple execution units must exist.

b) Panwar has not explicitly taught a plurality of multiplexers connected to the SP and PE resource files for selecting resource files from which the at least two execution units read data and to which the at least two execution units write data, a portion of the plurality of multiplexers associated with an execution unit controlled by a logical combination of the SP/PE bit and the CSB. However, Bapst has taught such a concept. See Fig.1, column 2, line 62, to column 3, line11, and the last paragraph of claim 7. Such a system allows a given execution unit to read from and write to one of multiple register files, and when operating in a second context, different register files may be read form and written to by the same given execution unit. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Panwar to include a plurality of multiplexers for selecting a register file to be read from and

written to. Note that the register file selected would be based on the current context, which is determined by the SP/PE bits and the CSB.

Response to Arguments

26. Applicant's arguments filed on October 12, 2004, have been fully considered but they are not persuasive.

27. Applicant argues the novelty/rejection of claim 14 on pages 12-13 of the remarks, in substance that:

> "Panwar does not teach and does not suggest a physical configuration of at least two processing elements. Panwar's system is not an array processor which supports multiple physical configurations. Panwar merely teaches a single processor that can be configured to operate as multiple virtual processors."

28. These arguments are not found persuasive for the following reasons:

a) Although Panwar has taught multiple virtual processors, these virtual processors operate using physical hardware resources of the overall processor. Without physical resources, processing cannot be performed. Therefore, the physical resources of the overall processor are configured such that multiple processing elements (virtual elements) may be realized.

29. Applicant argues the novelty/rejection of claim 14 on page 13 of the remarks, in substance that:

> "None of the bits in Gordon's status register, however, track the operating context of an array processor as presently claimed."

30. These arguments are not found persuasive for the following reasons:

a) The examiner did not state that the bits taught by Gordon anticipated applicant's claimed invention. Instead, Gordon was merely used to show that status bits may be stored in a register.

The bits that would be stored are the bits that would represent the state of each processing element, as shown in Fig.3. Clearly, each processor must know whether to operate, sleep, nap, etc. Therefore, bits must be used to represent these states. The examiner's position is that it would be obvious to store these status bits in a status register because Gordon has taught that status bits may be stored in a register.

31. Applicant argues the novelty/rejection of claim 14 on page 14 of the remarks, in substance that:

> "Panwar and Gordon do not teach that the first software task is written for the physical configuration and that the second software task is written for a second array processor having a different physical configuration."

32. These arguments are not found persuasive for the following reasons:

a) As disclosed in column 4, lines 15-27, of Panwar, if a first software task has N threads, then there will be a physical configuration of N virtual processing elements, whereas if a second software task has M threads, there will be a physical configuration of M virtual processing elements.

33. Applicant argues the novelty/rejection of claim 18 on pages 14-15 of the remarks, in substance that:

> "...the Office Action suggests that it is inherent that each processing element has a physical identifier. Applicants respectfully disagree. It cannot be assumed that each virtual processor of Panwar has both a virtual and physical identifier as claimed."

34. These arguments are not found persuasive for the following reasons:

a) Applicant has not claimed any specifics as to what the physical identifier (PID) identifies or represents. Consequently, the PID could be anything that identifies the "physicalness" of the

processing element in any way. For instance, the examiner had pointed out Fig.3 when saying that the physical identifier is inherent. To further explain, it can be seen from Fig.3 that each processing element has a state machine which represents its current physical state. Therefore, the states are physical IDs because they identify what physical state the corresponding element is in (active, sleeping, etc.).

35. Applicant argues the novelty/rejection of claims 24-27 and 29 on page 16 of the remarks, in substance that:

> "Panwar does not teach and does not suggest a merged VLIW SP and a VLIW PE processor which is configurable to operate in a first merged processor configuration or in a second merged processor configuration having a different physical configuration."

36. These arguments are not found persuasive for the following reasons:

a) Again, as discussed above in response to the arguments for claim 14, the SP and PE processors may be the virtual processing elements of Panwar. And, N virtual elements work together in different types of merged configurations to execute N threads in parallel. Keckler was used to teach the VLIW aspect of the invention.

37. Applicant argues the novelty/rejection of claim 24-27 and 29 on page 17 of the remarks, in substance that:

> "The IEU is neither an SP resource file nor a PE resource file as claimed. Furthermore, at the cited portion of Panwar, a copy of integer architectural register files if provided for each virtual processor. A copy of integer architectural register files is different than separate resource files such as the presently claimed SP resource file and the PE resource file."

38. These arguments are not found persuasive for the following reasons:

a) Applicant is correct in saying the IEU is not a register file. However, if the cited portion is reexamined (column 14, lines 57-63, and Fig.11), it can be realized that the IEU contains mechanisms to access the register files. And, in Fig.11, it can also be seen that there are multiple register files 1101, 1102, 1103, and 1104. That is, each virtual element has access to its own integer file and floating-point file. See column 15, lines 34-36.

39. Applicant argues the novelty/rejection of claim 24 on pages 18 of the remarks, in substance that:

> "Unlike each instruction carrying a thread ID, claim 24 claims an SP/PE bit carried in each instruction. In the present invention, the combination of the SP/PE bit carried in an instruction and the value of the CSB provide the basis upon which the merged VLIW SP/PE processor selects the merged processor configuration processing the instruction."

40. These arguments are not found persuasive for the following reasons:

a) Applicant has claimed no specifics regarding the SP/PE bit and the examiner asserts that a thread ID anticipates applicant's SP/PE bit. The thread ID will help determine the number of total threads that will be executed. In addition, the CSB is used to put each processing element in the appropriate state (active, sleep, napping...). Consequently, with both the thread ID (SP/PE) and the CSB of the prior art, a merged configuration may be realized because once the total number of threads is known, an equal number of virtual processors will be created (thereby forming a physical configuration of virtual processors). And, the CSB for each created virtual processor will be set so that they are in an active state (i.e. able to operate/execute). Clearly, the thread ID must be in the instruction, as is shown in Keckler's Fig.1, because when the instruction is fetched, the system must determine which virtual processor the receive the instruction for execution.

Conclusion

41. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (571) 272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

DJH
David J. Huisman
November 17, 2004

EDDIE CH
SUPERVISORY PAT
TECHNOLOGY C